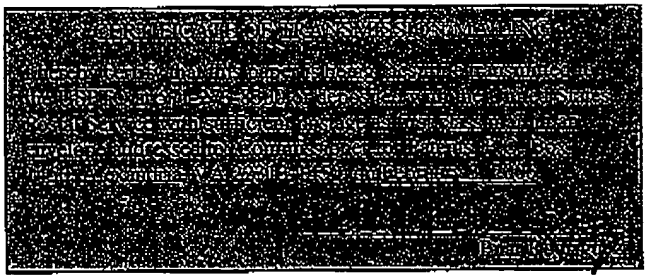
Patent

Attorney Docket No.: Intel 2207/6843
Serial No.: 09/470,875
Assignee: Intel Corporation

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Application No.	:	09/470,875 Confirmation No. 6722
Applicant	:	Manpreet S. KHAIRA
Filed	:	December 22, 1999
Title	:	METHOD AND APPARATUS FOR PERFORMING DISTRIBUTED SIMULATION UTILIZING A SIMULATION BACKPLANE
TC/A.U.	:	2123
Examiner	:	Dwin M. Craig
Customer No.:	:	25693

M/S Appeal Briefs - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450



ATTENTION: Board of Patent Appeals and Interferences

APPEAL BRIEF

Dear Sir:

This brief is in furtherance of the Notice of Appeal, filed in this case on October 6, 2005.

1. REAL PARTY IN INTEREST

The real party in interest in this matter is Intel Corporation. (Recorded May 8, 2000, Reel/Frame 010792/0621).

02/07/2006 LWONDIM1 00000005 110600 09470875
01 FC:1402 500.00 DA

2. RELATED APPEALS AND INTERFERENCES

There are no related appeals.

3. STATUS OF THE CLAIMS

Claims 1-56 are pending in the application. Claims 1, 2, 4, 5, 7-10, 13-18, 21, 22, 24-27, 29-32, 34,43, 45, 46, 48-51, and 53-56 are rejected under 35 U.S.C. §103(a) as being unpatentable over U.S. Patent No. 6,108,494 ("Eisenhofer-1"), in view of USP 5,881,270 ("Worthington"), and in further view of U.S. Patent No. 6,339,836 ("Eisenhofer-2"). Claims 3, 7, 20, 23, 28, 33, 36, 44, and 47 are rejected under 35 U.S. C. §103 (a) as being unpatentable over Eisenhofer-1 in view of Worthington and in further view of Eisenhofer-2, and in further view of U.S. Patent No. 6,175,946 ("Ly"). Claims 11 and 12 are rejected under 35 U.S. C. §103 (a) as being unpatentable over Eisenhofer-1 in view of Worthington and in further view of Eisenhofer-2 and in further view of U.S. Patent No. 5,881,267 ("Dearth-1"). Claim 19 is rejected under 35 U.S. C. §103 (a) as being unpatentable over Eisenhofer-1 in view of Worthington and in further view of Eisenhofer-2, and in further view of U.S. Patent No. 5,732,247 ("Dearth-2"). Claims 11 and 12 are rejected under 35 U.S. C. §103 (a) as being unpatentable over Eisenhofer-1 in view of Worthington and in further view of Eisenhofer-2 and in further view of Dearth-2.

4. STATUS OF AMENDMENTS

Applicants did not make any amendments to the claim subsequent to final rejection. The claims listed on page 1 of the Appendix attached to this Appeal Brief reflect the present status of the claims (including amendments entered after final rejection).

5. SUMMARY OF THE CLAIMED SUBJECT MATTER

The embodiment of claim 1 generally describes a method for performing distributed simulation, comprising: providing at least two simulators (e.g., *see* page 6, line 5, Figure 1, 106, 108), wherein at least one of the at least two simulators represents at least one of a component and a system based on processors and chipsets (e.g., *see* page 5, line 5-7, Figure 1); providing a backplane having a fixed configuration (e.g., *see* page 5, line 5-7, Figure 1, 102); associating an interface with each of the at least two simulators (e.g., *see* page 5, lines 7-8, Figure 1, 110); interfacing each of the at least two simulators with the fixed configuration backplane via the interface associated with each of the at least two simulators (e.g., *see* page 5, lines 7-8); exchanging messages between the at least two simulators via the fixed configuration backplane and the associated interfaces (e.g., *see* page 6, lines 13-15); and operating each interface to convert the messages between a data format associated with the fixed configuration backplane and a data format associated with the simulator associated with the interface (e.g., *see* page 6, lines 15-17).

The embodiment of claim 21 generally describes a method for performing distributed simulation, comprising: providing at least two simulators (e.g., *see* page 6, line 5, Figure 1, 106, 108), wherein at least one of the at least two simulators represents at least one of a component and a system based on microprocessors and complex chipsets (e.g., *see* page 5, line 5-7, Figure 1); associating an interface with each of the at least two simulators (e.g., *see* page 5, lines 7-8, Figure 1, 110); interfacing each of the at least two simulators with a simulation backplane via the interface associated with each of the at least two simulators without performing a reconfiguration of the backplane (e.g., *see* page 5, lines 7-8); exchanging messages between the at least two

simulators via the simulation backplane and the associated interfaces (e.g., *see* page 6, lines 13-15); changing a combination of the at least two simulators and associated interfaces without performing a reconfiguration of the simulation backplane (e.g., *see* page 5, lines 12-14); wherein each interface converts messages between a data format associated with the backplane and a data format associated with the simulator associated with the interface (e.g., *see* page 6, lines 15-17).

The embodiment of claim 26 generally describes a method for validating a component, comprising: sending a test message from a backplane to an interface (e.g., *see* page 6, lines 13-15); converting the test message from a first data format into a second data format , utilizing the interface (e.g., *see* page 6, lines 15-17); sending the converted test message from the interface to at least one of the component and at least one model of the component (e.g., *see* page 8, lines 6-7, Figure 4, step 406); receiving, at the interface and in response to the converted test message, a response message from at least one of the component and the at least one model of the component, the response message being in the second data format (e.g., *see* page 8, lines 7-9, Figure 4, steps 408 & 410); converting the response message from the second data format to the first data format, utilizing the interface; sending the converted response message to the backplane (e.g., *see* page 6, lines 10-11); and comparing the converted response message to a predetermined value (e.g., *see* page 6, lines 11-13).

The embodiment of claim 29 generally describes a method for validating a component, comprising: receiving a test message from a first component; sending the test message to at least one of the component and at least one model of the component via at least one interface (e.g., *see* page 6, lines 13-15) which converts the test message from a first data format to a second data format (e.g., *see* page 6, lines 15-17); receiving a response message from at least one of the component and the at least one model of the component via the at least one interface which

converts the response message from the second data format to the first data format (e.g., *see* page 8, lines 7-9, Figure 4, steps 408 & 410); sending the response message to a second device (e.g., *see* page 6, lines 10-11); and comparing the response message to a predetermined value utilizing the second device (e.g., *see* page 6, lines 11-13).

The embodiment of claim 34 generally describes a method for validating a component, comprising: receiving a test message from a backplane (e.g., *see* page 6, lines 13-15); converting the test message from a first data format to a second data format (e.g., *see* page 6, lines 15-17); sending the test message to at least one of the component and at least one model of the component (e.g., *see* page 8, lines 6-7, Figure 4, step 406); receiving, in response to the test message, a response message from at least one of the component and the at least one model of the component; converting the response message from the second data format to the first data format (e.g., *see* page 8, lines 7-9, Figure 4, steps 408 & 410); sending the response message to a first device via a backplane (e.g., *see* page 6, lines 10-11); and comparing the response message to a predetermined value utilizing the first device(e.g., *see* page 6, lines 11-13).

The embodiment of claim 40 generally describes an apparatus for validating a component, comprising: a processor; a computer readable memory segment adapted to be connected to said processor; an interface module stored within the computer readable memory segment (e.g., *see* page 5, lines 7-8, Figure 1, 110); a simulation backplane module within the computer readable memory segment (e.g., *see* page 5, line 5-7, Figure 1, 102), the simulation backplane module comprising computer program code segments which, when executed by the processor, implement the steps of: receiving a test message from a first device (e.g., *see* page 6, lines 13-15); sending the test message to at least one of the component and at least one model of the component via the interface module which converts the test message from a first data format

to a second data format (e.g., *see* page 8, lines 6-7, Figure 4, step 406); receiving a response message from at least one of the component and the at least one model of the component via the interface module which converts the response message from the second data format to the first data format (e.g., *see* page 8, lines 7-9, Figure 4, steps 408 & 410); and sending the response message to a second device (e.g., *see* page 6, lines 10-11), the second device comparing the response message to a predetermined value (e.g., *see* page 6, lines 11-13).

The embodiment of claim 45 generally describes an apparatus for validating a component, comprising: a processor; a computer readable memory segment adapted to be connected to said processor; a simulation backplane module within the computer readable memory segment (e.g., *see* page 5, lines 7-8, Figure 1, 110); an interface module stored within the computer readable memory segment (e.g., *see* page 5, line 5-7, Figure 1, 102), the interface module comprising computer program code segments which, when executed by said processor, implement the steps of: receiving a test message from the simulation backplane module (e.g., *see* page 6, lines 13-15); converting the test message from a first data format to a second data format; sending the test message to at least one of the component and at least one model of the component (e.g., *see* page 8, lines 6-7, Figure 4, step 406); receiving, in response to the test message, a response message from at least one of the component and the at least one model of the component; converting the response message from the second data format to the first data format (e.g., *see* page 8, lines 7-9, Figure 4, steps 408 & 410); and sending the response message to a first device via the simulation backplane module (e.g., *see* page 6, lines 10-11), the first device comparing the response message to a predetermined value (e.g., *see* page 6, lines 11-13).

The embodiment of claim 51 generally describes a method for performing distributed simulation, comprising: providing at least two simulators (e.g., *see* page 6, line 5, Figure 1, 106,

108), wherein at least one of the at least two simulators represents at least one of a component and a system based on processors and chipsets (e.g., *see* page 5, line 5-7, Figure 1, 102); associating an interface with each of the at least two simulators (e.g., *see* page 5, line 7-8, Figure 1, 110); interfacing each of the at least two simulators with a simulation backplane via the interface associated with each of the at least two simulators without performing a reconfiguration of the backplane (e.g., *see* page 5, lines 7-8); and exchanging messages between the at least two simulators via the simulation backplane and the associated interfaces (e.g., *see* page 6, lines 13-15); wherein each interface converts messages between a data format associated with the backplane and a data format associated with the simulator associated with the interface (e.g., *see* page 6, lines 15-17).

The embodiment of claim 52 generally describes an article, comprising: a storage medium having stored thereon instructions when executed by a processor cause a system to validate a component by performing the following steps: providing at least two simulators (e.g., *see* page 6, line 5, Figure 1, 106, 108), wherein at least one of the at least two simulators represents at least one of a component and a system based on processors and chipsets; providing a backplane having a fixed configuration (e.g., *see* page 5, line 5-7, Figure 1, 102); associating an interface with each of the at least two simulators (e.g., *see* page 5, line 7-8, Figure 1, 110); interfacing each of the at least two simulators with the fixed configuration backplane via the interface associated with each of the at least two simulators (e.g., *see* page 5, lines 7-8); exchanging messages between the at least two simulators via the fixed configuration backplane and the associated interfaces (e.g., *see* page 6, lines 13-15); and operating each interface to convert the messages between a data format associated with the fixed configuration backplane

and a data format associated with the simulator associated with the interface (e.g., *see* page 6, lines 15-17).

The embodiment of claim 53 generally describes a article, comprising: a storage medium having stored thereon instructions when executed by a processor cause a system to validate a component by performing the following steps: receiving a test message from a first component (e.g., *see* page 6, lines 13-15); sending the test message to at least one of the component and at least one model of the component via at least one interface (e.g., *see* page 8, lines 6-7, Figure 4, step 406) which converts the test message from a first data format to a second data format; receiving a response message from at least one of the component and the at least one model of the component via the at least one interface which converts the response message from the second data format to the first data format (e.g., *see* page 8, lines 7-9, Figure 4, steps 408 & 410); sending the response message to a second device (e.g., *see* page 6, lines 10-11); and comparing the response message to a predetermined value utilizing the second device (e.g., *see* page 6, lines 11-13).

The embodiment of claim 54 generally describes an article, comprising: a storage medium having stored thereon instructions when executed by a processor cause a system to validate a component by performing the following steps: receiving a test message from a backplane (e.g., *see* page 6, lines 13-15); converting the test message from a first data format to a second data format (e.g., *see* page 8, lines 6-7, Figure 4, step 406); sending the test message to at least one of the component and at least one model of the component; receiving, in response to the test message, a response message from at least one of the component and the at least one model of the component; converting the response message from the second data format to the first data format (e.g., *see* page 8, lines 7-9, Figure 4, steps 408 & 410); sending the response message to a

first device via a backplane (e.g., *see* page 6, lines 10-11); and comparing the response message to a predetermined value utilizing the first device (e.g., *see* page 6, lines 11-13).

The embodiment of claim 55 generally describes an article, comprising: a storage medium having stored thereon instructions when executed by a processor cause a system to validate a component by performing the following steps: sending a test message from a backplane to an interface (e.g., *see* page 6, lines 13-15); converting the test message from a first data format into a second data format, utilizing the interface (e.g., *see* page 8, lines 6-7, Figure 4, step 406); sending the converted test message from the interface to at least one of the component and at least one model of the component; receiving, at the interface and in response to the converted test message, a response message from at least one of the component and the at least one model of the component, the response message being in the second data format; converting the response message from the second data format to the first data format, utilizing the interface (e.g., *see* page 8, lines 7-9, Figure 4, steps 408 & 410); sending the converted response message to the backplane (e.g., *see* page 6, lines 10-11); and comparing the converted response message to a predetermined value (e.g., *see* page 6, lines 11-13).

The arrangement of the present invention provides a comprehensive solution for system, component and design validation. A typical distributed simulation involves running many simulators corresponding to different components as independent processes. Fig. 1 shows a validation environment comprising a simulation backplane, four simulators, and four SDI's according to an embodiment of the present invention. In Fig. 1, a driver 104, a checker 112, a C model 108 of an integrated circuit chip and a simulator (CSIM) model 106 of an integrated circuit chip are interfaced with a simulation backplane 102 via SDI's 110. Each of the components shown in Fig. 1 correspond to algorithms which may be implemented as computer

programs in any programming language such as C++. These computer programs may be executed on one workstation, a cluster of workstations or on highly-parallel computers such as the Paragon™ (available from Intel Corporation, Santa Clara, California).

CSIM model 106 and C model 108 are simulators which mimic an integrated circuit chip at different levels of abstraction. Backplane 102 connects these two models of the integrated circuit chip. During the process of performing a distributed simulation, data messages representing input stimuli from the driver 104 are forwarded by the simulation backplane 102 to the models 106 and 108. The models 106 and 108 operate on the data and return response messages to the simulation backplane 102. The simulation backplane 102 then forwards the response messages to checker 112. Checker 112 then determines whether the models performed as expected by comparing the received response messages with predetermined values. The driver 104, the models 106 and 108, and the checker 112 exchange messages with the simulation backplane 102 via corresponding SDI's 110. Each SDI converts the exchanged messages between the data format supported by its corresponding simulator and the data format supported by the simulation backplane. Thus, simulation backplane 102 may implement a common data format when communicating with any simulator, driver, or checker.

Fig. 2 shows an apparatus suitable for performing distributed simulation according to an embodiment of the present invention.

Fig. 3 shows the apparatus of Fig. 2 modified to implement the validation environment of Fig. 1 according to an embodiment of the present invention. In addition to the corresponding elements described above with respect to Fig. 2, metasimulator 301 of Fig. 3 includes a CSIM model module 306, a C model module 310, a driver module 314, and a checker interface module 320 which respectively correspond to the similarly named simulators of Fig. 1. Metasimulator

301 also includes the following modules which correspond to the SDI's 110 of Fig. 1: CSIM interface module 308, C mode interface module 312, driver interface module 316, and checker interface module 320. The interface modules interface their similarly named simulator modules with simulation backplane module 322. When executed by processor 302, the modules of Fig. 3 implement the distributed simulation method of the present invention in a manner similar to that described above with respect to Fig. 1.

Fig. 4 shows a flow chart with an example of the steps which may be implemented by an SDI according to an embodiment of the present invention. These steps may be stored, for example, as computer program code segments within interface module 208 of Fig. 2 or within any of the interface modules of Fig. 3. In step 402, the SDI receives a test message from a backplane such as backplane 102 of Figure 1 (or one of the backplane modules shown in either Fig. 2 or Fig. 3). In step 404, the SDI converts the test message from the backplane's data format to a format suitable for the simulator this particular SDI is interfacing with the backplane. In step 406, the SDI sends the converted test message to the simulator. In step 408, the SDI receives a response message from the simulator. In step 410, the SDI converts the response message to the backplane's data format. In step 412, the SDI sends the converted test message to the backplane.

6. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

A. Are claims 1, 2, 4, 5, 7-10, 13-18, 21, 22, 24-27, 29-32, 34,43, 45, 46, 48-51, and 53-56 rendered obvious under 35 U.S.C. §103(a) by Eisenhofer et al. (U.S. Patent No. 6,108,494) ("Eisenhofer-1"), in view of Worthington et al. (U.S. Patent No. 5,881,270)

("Worthington"), and in further view of Eisenhofer et al. (U.S. Patent No. 6,339,836) ("Eisenhofer-2")?

7. ARGUMENT

A. Claims 1, 2, 4, 5, 7-10, 13-18, 21, 22, 24-27, 29-32, 34,43, 45, 46, 48-51, and 53-56 are not rendered obvious by Eisenhofer-1, Worthington, and Eisenhofer-2

Applicants respectfully submit the cited reference do not teach , suggest or disclose at least "[a] method for performing distributed simulation, comprising: ...associating an interface with each of the at least two simulators ... and operating each interface to convert the messages between a data format associated with the fixed configuration backplane and a data format associated with the simulator associated with the interface" (e.g., as described in claim 1).

The Examiner asserts that Eisenhofer-1 teaches "data format conversions" at column 5 lines 52-67, column 6 lines 1-20, column 12 lines 34-67, and column 13 lines 1-5 and "an interface for the simulators" at column 5 lines 52-67 and column 6 lines 1-20. Applicants disagree. Column 6, lines 1-20 state:

> FIG. 2 conceptually illustrates the N-way co-simulation paradigm. A simulation backplane 210, such as the SimMatrix electrical simulation backplane available from Mentor Graphics Corporation, is coupled to a plurality of simulators 231-234 through their respective interfaces 241-244. The simulator interfaces 241-244 each expose a standard set of interface routines for performing functions such as initializing the simulator, performing netlist generation, e.g., writing cell descriptions, parsing netlists assigned to the simulator, registering boundary events, performing data type conversions, notifying the simulation backplane 210 when boundary events occur, and other functions to facilitate simulation. *The simulation backplane 210 defines the simulation environment 200 by way of a set of procedures, protocols and functions and coordinates the interaction among the simulators 231-234. For example, the simulation backplane 210 coordinates event transaction processing, state translation, event arbitration, and manages synchronization processing among the simulators 231-234.* Synchronization is the point in the simulation session at which the simulators 231-234 and the simulation backplane 210 agree on the value of time. It is only during synchronization that boundary event information, such as states, currents, or voltages, can be reliably exchanged between simulators. The frequency of synchronizations during a simulation session is directly related to accuracy and performance; the more frequent the synchronizations, the

> higher the resulting accuracy (to a limit) and the lower the performance (without limit). *When a boundary event occurs between simulators, the simulation backplane 210 synchronizes the simulators so that they are at the same point in time and, before transferring any event information, it converts the event information to a representation usable by the target simulator.* In this manner, data is reliably exchanged between simulators. (*emphasis added)*

Applicants submit column 5 lines 52-67 and column 6 lines 1-20 of Eisenhofer-1 disclose a simulation backplane 210 coupled to a plurality of simulators 231-234 through their respective interfaces 241-244. The simulation backplane 210 coordinates event transaction processing, state translation, event arbitration, and manages synchronization processing among the simulators 231-234. None of these functions performed by the simulation backplane 210 – event transaction processing, state translation etc., however, is the equivalent of operating *each interface to convert* the messages between a data format associated with the fixed configuration backplane and a data format associated with the simulator associated with the interface as described in embodiments of the present application.

As for the Examiner's claim that column 5 lines 52-67 and column 6 lines 1-20 disclose "an interface for the simulators", it is true the Eisenhofer-1 reference discloses the simulators 231-234 and their respective interfaces 241-244. However, the sections describing the simulators or their interfaces do not describe operating *each interface to convert* the messages between a data format associated with the fixed configuration backplane and a data format associated with the simulator associated with the interface as described in embodiments of the present application.

The Examiner further asserts that column 12 lines 40-41 disclose that the simulation backplane performs data type conversion. Column 12 line 40-41 state:

> In addition to synchronizing the simulators, the simulation backplane 210 also performs data type conversion and transfers boundary event information, such as signal state.

Indeed, Applicants agree. In Eisenhofer-1, *the simulation backplane* performs a data type conversion. However, as argued above, in embodiments of the present application it is not the *simulation backplane* that performs a conversion messages between a data format associated with the fixed configuration backplane and a data format associated with the simulator associated with the interface, but rather *the interfaces* associated with the simulators. Eisenhofer-1 *does not* describe such a limitation.

Other sections of Eisenhofer-1 confirm this. The cited section column 12 lines 34-61 states: "[t]herefore, before transferring the signal state from the source simulator to one or more target simulators, signal mapping (also referred to as data type conversion) is performed between the source and target representations in order to achieve consistent signal state representations. Exemplary conversions include the point-to-point signal mapping or mapping to intermediate simulation backplane types... Either conversion mechanism may be implemented with *a user-programmable state translation table*" (*emphasis added*). Column 6 line 15 of Eisenhofer-1 describes: "[w]hen a boundary event occurs between simulators, *the simulation backplane 210* synchronizes the simulators so that they are at the same point in time and, before transferring any event information, *it converts* the event information to a representation usable by the target simulator" (*emphasis added*). Furthermore, column 12 line 4 states: "[i]n addition to synchronizing the simulators, *the simulation backplane 210* also performs data type conversion and transfers boundary event information, such as signal state" (*emphasis added*). Therefore, the Eisenhofer-1 reference discloses performing data type conversion with a simulation backplane (possibly in conjunction with a user-programmable state translation table) in multiple embodiments.

However, Applicants respectfully submit that none of the cited sections of Eisenhofer-1 teach, suggest or disclose "...*operating each interface to convert the messages between a data format associated with the fixed configuration backplane* and a data format associated with the simulator associated with the interface" as specifically described in the embodiment of independent claim 1. Instead, as discussed above, the conversion between data types is accomplished with the use of the simulation backplane or a user-programmable state translation table in Eisenhofer-1.

Eisenhofer-2 fails to make up for the deficiencies of Eisenhofer-1. Eisenhofer-2 is directed towards design partitioning, and although it employs the use of a simulation backplane, simulators and simulator interfaces, it does not disclose at least operating each interface to convert the messages between a data format associated with the fixed configuration backplane.

Worthington fails to support a proper §103(a) rejection for similar reasons. Worthington is directed toward flexible simulation modeling of multi-component systems using global mailboxes, and does not disclose at least operating each interface to convert the messages between a data format associated with the fixed configuration backplane as described in embodiments of the present application.

Similarly, Applicants submit there is no disclosure of operating each interface to convert the messages between a data format associated with the fixed configuration backplane in the Ly, Dearth-1, and Dearth-2 references as well.

Therefore, since the specific limitations "operating each interface to convert the messages between a data format associated with the fixed configuration backplane and a data format associated with the simulator associated with the interface" are not taught or suggested anywhere in any of the cited references, claim 1 is in condition for allowance and the 35 U.S.C. 103(a)

rejection should be withdrawn. Applicants respectfully submit that independent claims 21, 26, 29, 34, 40, 45, 51, 52, 53, 54, and 55 also contain the allowable limitations similar to the one described above, and are therefore are allowable for similar reasons. Furthermore, Applicants submit that dependent claims 2-20, 22-25, 27-28, 30-33, 35-39, 41-44, and 46-50 are allowable as depending from allowable base claims.

For at least these reasons, the Claims 1-56 are believed to be patentable over the cited references, individually and in combination. Withdrawal of the rejections is, therefore, respectfully requested.

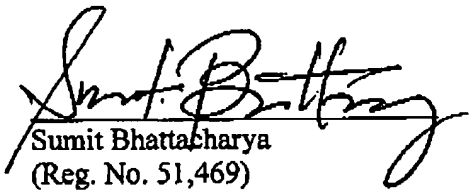
Appellants therefore respectfully request that the Board of Patent Appeals and Interferences reverse the Examiner's decision rejecting claims 1-18 and direct the Examiner to pass the case to issue.

The Examiner is hereby authorized to charge the appeal brief fee of **$500.00** and any additional fees which may be necessary for consideration of this paper to Kenyon & Kenyon Deposit Account No. **11-0600**.

Respectfully submitted,

KENYON & KENYON LLP

Date: February 3, 2006

By: 
Sumit Bhattacharya
(Reg. No. 51,469)

KENYON & KENYON LLP
333 West San Carlos St., Suite 600
San Jose, CA 95110
Telephone: (408) 975-7500
Facsimile: (408) 975-7501

APPENDIX

(Brief of Appellants Manpreet S. KHAIRA et al.
U.S. Patent Application Serial No. 09/470,875)

8. CLAIMS ON APPEAL

1. A method for performing distributed simulation, comprising:

providing at least two simulators, wherein at least one of the at least two simulators represents at least one of a component and a system based on processors and chipsets;

providing a backplane having a fixed configuration;

associating an interface with each of the at least two simulators;

interfacing each of the at least two simulators with the fixed configuration backplane via the interface associated with each of the at least two simulators;

exchanging messages between the at least two simulators via the fixed configuration backplane and the associated interfaces; and

operating each interface to convert the messages between a data format associated with the fixed configuration backplane and a data format associated with the simulator associated with the interface.

2. The method of claim 1, wherein at least one of the at least two simulators is a driver.

3. The method of claim 1, wherein at least one of the at least two simulators is a checker.

4. The method of claim 1, wherein at least one of the at least two simulators is a model.

5. The method of claim 1, wherein the component is an integrated circuit chip.

7. The method of claim 1, wherein communication between the at least two simulators is based upon one of a central controlling process and a tree.

8. (Cancelled)

9. The method of claim 1, wherein only the simulators requiring relaxation are relaxed prior to exchanging messages.

10. The method of claim 1, wherein a system comprising both cycle based simulators and event based simulators is relaxed by holding the cycle based simulators at a designated clock edge until the cycle based simulators are relaxed and by incrementally advancing time along a cycle from the designated clock edge until the event-based simulators are relaxed, the cycle based simulators and the event based simulators being relaxed before advancing a simulation time.

11. The method of claim 1, further comprising:

executing a deadlock-free remote procedure call function.

12. The method of claim 11, further comprising:

adding new signals to the backplane global signal vector utilizing the remote procedure call capability.

13. The method of claim 1, further comprising:

executing multiple-bit encodings of the exchanged messages to enable simulators based on different encoding schemes to communicate via the simulation backplane.

14. The method of claim 1, further comprising:

resolving a conflict between a signal strength of messages exchanged via the simulation backplane utilizing a least upper bound operation accomplished by a bitwise-OR operation.

15. The method of claim 1, wherein messages are exchanged in a manner which accommodate both word-parallel mode capable simulators and non-word-parallel mode capable simulators.

16. The method of claim 1, further comprising:

executing a protocol such that a simulation backplane least upper bound operation preserves enough information to enable each interface to locally determine and correctly resolve a value of messages exchanged via the simulation backplane.

17. The method of claim 1, further comprising:

exchanging messages in a manner which accommodate simulators running at different frequencies.

18. The method of claim 1, wherein design validation tests can be written using High-Level Language (HLL) primitives.

19. The method of claim 1, wherein one of a same handwritten test and an ATG generated test can be used for all simulators.

20. The method of claim 1, wherein a protocol checker may be used with an arbitrary collection of simulators.

21. A method for performing distributed simulation, comprising:

providing at least two simulators, wherein at least one of the at least two simulators represents at least one of a component and a system based on microprocessors and complex chipsets;

associating an interface with each of the at least two simulators;

interfacing each of the at least two simulators with a simulation backplane via the interface associated with each of the at least two simulators without performing a reconfiguration of the backplane;

exchanging messages between the at least two simulators via the simulation backplane and the associated interfaces;

changing a combination of the at least two simulators and associated interfaces without performing a reconfiguration of the simulation backplane;

wherein each interface converts messages between a data format associated with the backplane and a data format associated with the simulator associated with the interface.

22. The method of claim 21, wherein at least one of the at least two simulators is a driver.

23. The method of claim 21, wherein at least one of the at least two simulators is a checker.

24. The method of claim 21, wherein at least one of the at least two simulators is a model.

25. The method of claim 21, wherein the component is an integrated circuit chip.

26. A method for validating a component, comprising:

sending a test message from a backplane to an interface;

converting the test message from a first data format into a second data format, utilizing the interface;

sending the converted test message from the interface to at least one of the component and at least one model of the component;

receiving, at the interface and in response to the converted test message, a response message from at least one of the component and the at least one model of the component, the response message being in the second data format;

converting the response message from the second data format to the first data format, utilizing the interface;

sending the converted response message to the backplane; and

comparing the converted response message to a predetermined value.

27. The method of claim 26, wherein the component is an integrated circuit chip.

28. The method of claim 26, wherein the comparing step is accomplished utilizing a checker.

29. A method for validating a component, comprising:

receiving a test message from a first component;

sending the test message to at least one of the component and at least one model of the component via at least one interface which converts the test message from a first data format to a second data format;

receiving a response message from at least one of the component and the at least one model of the component via the at least one interface which converts the response message from the second data format to the first data format;

sending the response message to a second device; and

comparing the response message to a predetermined value utilizing the second device.

30. The method of claim 29, wherein the component is an integrated circuit chip.

31. The method of claim 29, wherein the first device is a driver.

32. The method of claim 29, wherein the first device is one of another component and a model of another component.

33. The method of claim 29, wherein the second device is a checker.

34. A method for validating a component, comprising:

receiving a test message from a backplane;

converting, utilizing an interface, the test message from a first data format to a second data format;

sending the test message to at least one of the component and at least one model of the component;

receiving, in response to the test message, a response message from at least one of the component and the at least one model of the component;

converting, using an interface, the response message from the second data format to the first data format;

sending the response message to a first device via a backplane; and

comparing the response message to a predetermined value utilizing the first device.

35. The method of claim 34, wherein the component is an integrated circuit chip.

36. The method of claim 34, wherein the first device is a checker.

37. The method of claim 34, wherein the test message is received from a second device.

38. The method of claim 37, wherein the second device is a driver.

39. The method of claim 37, wherein the second component is one of another integrated circuit chip and a model of another integrated circuit chip.

40. An apparatus for validating a component, comprising:

a processor;

a computer readable memory segment adapted to be connected to said processor;

an interface module stored within the computer readable memory segment;

a simulation backplane module within the computer readable memory segment, the simulation backplane module comprising computer program code segments which, when executed by the processor, implement the steps of:

receiving a test message from a first device;

sending the test message to at least one of the component and at least one model of the component via the interface module which converts the test message from a first data format to a second data format;

receiving a response message from at least one of the component and the at least one model of the component via the interface module which converts the response message from the second data format to the first data format; and

sending the response message to a second device, the second device comparing the response message to a predetermined value.

41. The apparatus of claim 40, wherein the component is an integrated circuit chip.

42. The apparatus of claim 40, wherein the first device is a driver.

43. The apparatus of claim 40, wherein the first device is one of another component and a model of another component.

44. The method of claim 40, wherein the second device is a checker.

45. An apparatus for validating a component, comprising:

a processor;

a computer readable memory segment adapted to be connected to said processor;

a simulation backplane module within the computer readable memory segment;

an interface module stored within the computer readable memory segment, the interface module comprising computer program code segments which, when executed by said processor, implement the steps of:

receiving a test message from the simulation backplane module;

converting the test message from a first data format to a second data format;

sending the test message to at least one of the component and at least one model of the component;

receiving, in response to the test message, a response message from at least one of the component and the at least one model of the component;

converting the response message from the second data format to the first data format; and

sending the response message to a first device via the simulation backplane module, the first device comparing the response message to a predetermined value.

46. The apparatus of claim 45, wherein the component is an integrated circuit chip.

47. The apparatus of claim 45, wherein the first component is a checker.

48. The method of claim 45, wherein the test message is received from a second device.

49. The method of claim 48, wherein the second device is a driver.

50. The method of claim 48, wherein the second device is one of another component and a model of another component.

51. A method for performing distributed simulation, comprising:

providing at least two simulators, wherein at least one of the at least two simulators represents at least one of a component and a system based on processors and chipsets;

associating an interface with each of the at least two simulators;

interfacing each of the at least two simulators with a simulation backplane via the interface associated with each of the at least two simulators without performing a reconfiguration of the backplane; and

exchanging messages between the at least two simulators via the simulation backplane and the associated interfaces;

wherein each interface converts messages between a data format associated with the backplane and a data format associated with the simulator associated with the interface.

52. An article, comprising:

a storage medium having stored thereon instructions when executed by a processor cause a system to validate a component by performing the following steps:

providing at least two simulators, wherein at least one of the at least two simulators represents at least one of a component and a system based on processors and chipsets;

providing a backplane having a fixed configuration;

associating an interface with each of the at least two simulators;

interfacing each of the at least two simulators with the fixed configuration backplane via the interface associated with each of the at least two simulators;

exchanging messages between the at least two simulators via the fixed configuration backplane and the associated interfaces; and

operating each interface to convert the messages between a data format associated with the fixed configuration backplane and a data format associated with the simulator associated with the interface.

53. An article, comprising:

a storage medium having stored thereon instructions when executed by a processor cause a system to validate a component by performing the following steps:

receiving a test message from a first component;

sending the test message to at least one of the component and at least one model of the component via at least one interface which converts the test message from a first data format to a second data format;

receiving a response message from at least one of the component and the at least one model of the component via the at least one interface which converts the response message from the second data format to the first data format;

sending the response message to a second device; and

comparing the response message to a predetermined value utilizing the second device.

54. An article, comprising:

a storage medium having stored thereon instructions when executed by a processor cause a system to validate a component by performing the following steps:

receiving a test message from a backplane;

converting, utilizing an interface, the test message from a first data format to a second data format;

sending the test message to at least one of the component and at least one model of the component;

receiving, in response to the test message, a response message from at least one of the component and the at least one model of the component;

converting, utilizing an interface, the response message from the second data format to the first data format;

sending the response message to a first device via a backplane; and

comparing the response message to a predetermined value utilizing the first device.

55. An article, comprising:

a storage medium having stored thereon instructions when executed by a processor cause a system to validate a component by performing the following steps:

sending a test message from a backplane to an interface;

converting the test message from a first data format into a second data format, utilizing the interface;

sending the converted test message from the interface to at least one of the component and at least one model of the component;

receiving, at the interface and in response to the converted test message, a response message from at least one of the component and the at least one model of the component, the response message being in the second data format;

converting the response message from the second data format to the first data format, utilizing the interface;

sending the converted response message to the backplane; and

comparing the converted response message to a predetermined value.

56. The method of claim 1, wherein exchanged messages are gathered together into a global signal vector.

9. EVIDENCE APPENDIX

No further evidence has been submitted with this Appeal Brief.

10. RELATED PROCEEDINGS APPENDIX

Per Section 2 above, there are no related proceedings to the present Appeal.